Software Defined Network (SDN)

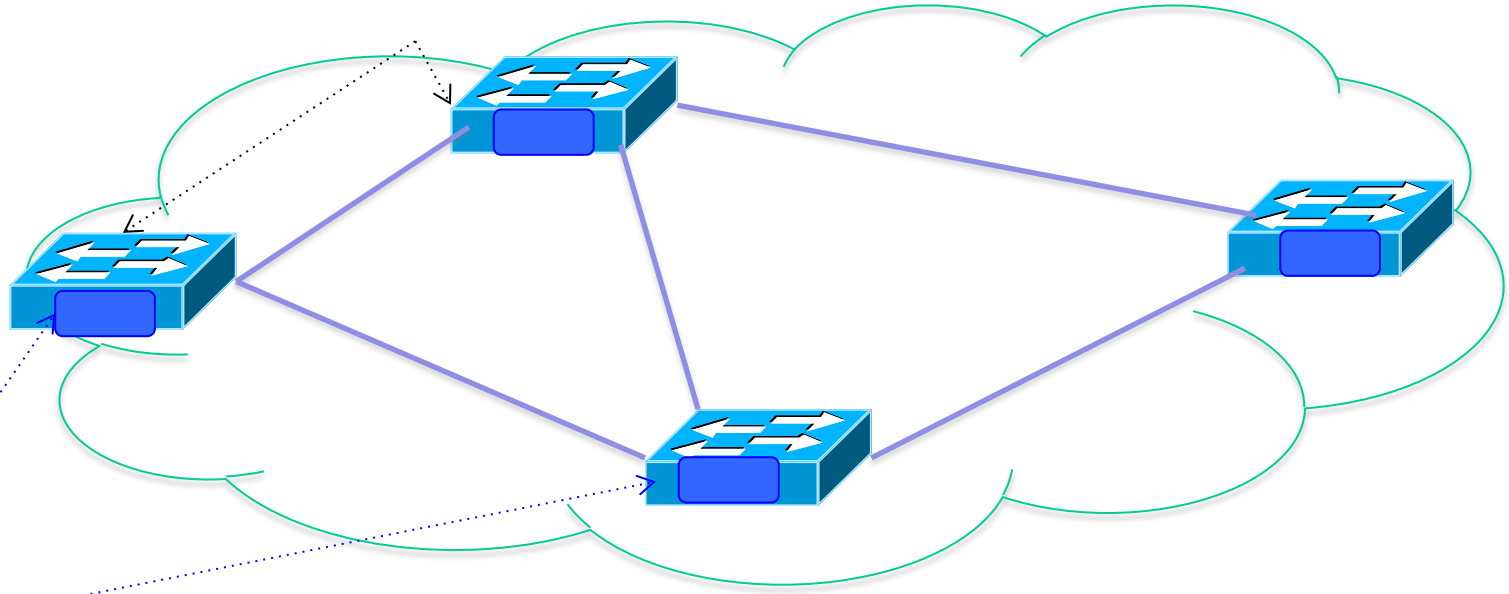
agenda

- Introduction to SDN
- Brief introduction to the Openflow protocol

INSA de Toulouse • 135, Avenue de Rangueil • 31077 Toulouse
Tél. : +33(0)5.61.55.95.13 • Fax : +33(0)5.61.55.95.00 • www.insa-toulouse.fr

1. Introduction to SDN

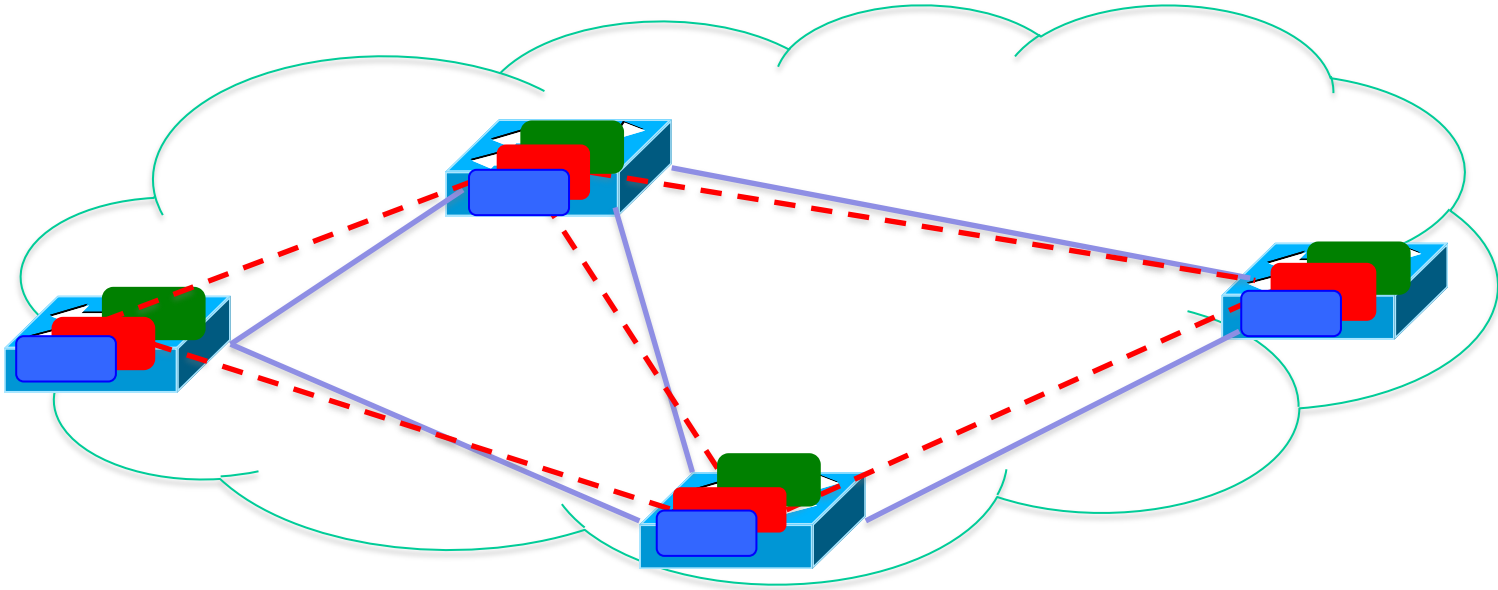
- **Positioning w.r.t legacy computer networks:**
  - legacy computer networks



- **Data plane: network functions that handle packets**
  - forwarding,marking,shape/police traffic and measure&count packets

SDN vs legacy computer networks

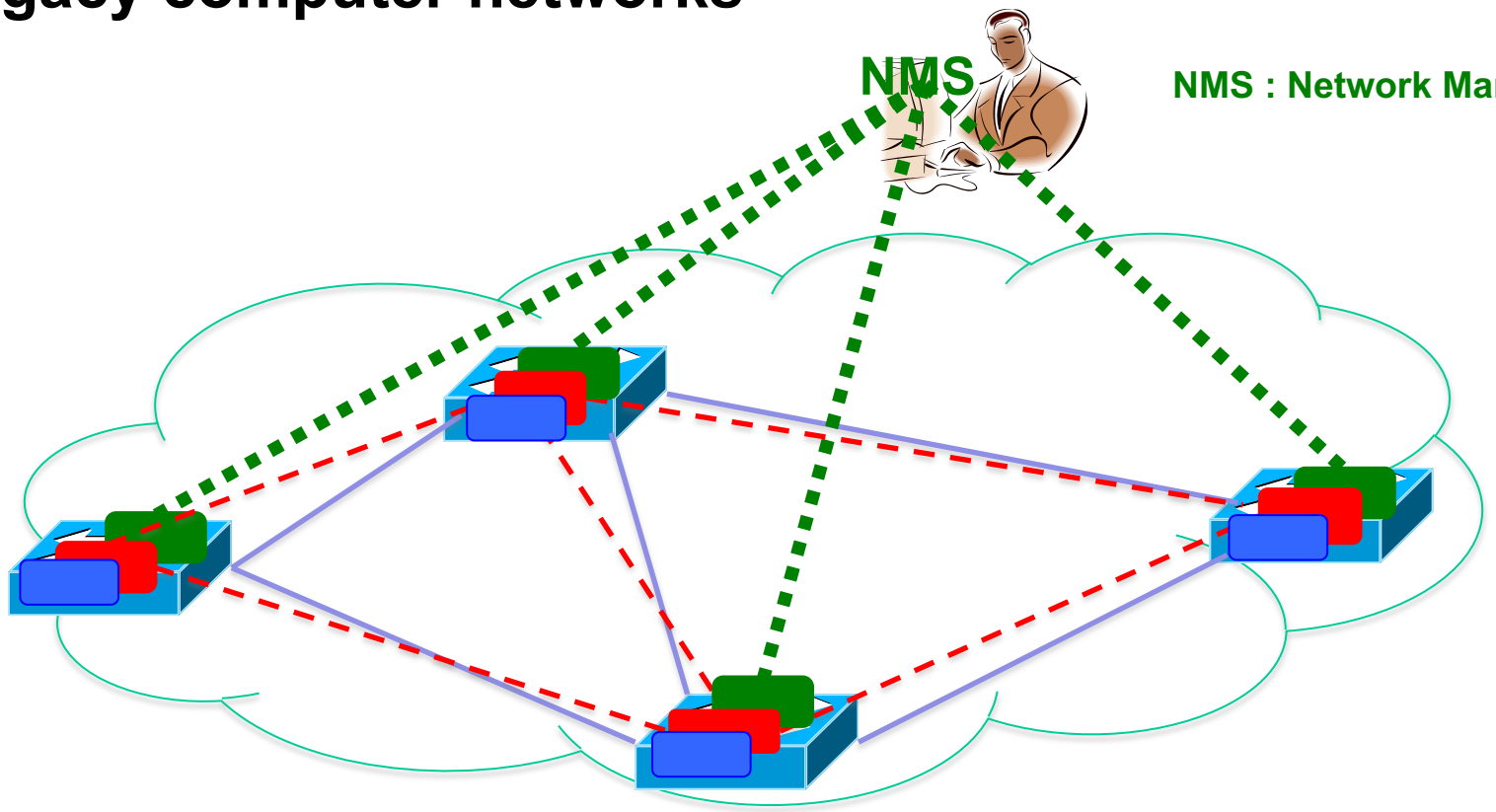
- **Legacy computer networks**



- **Control plane: network Functions that specify how packets are treated at each node**
  - **distributed algorithms** that track network topology evolutions, compute and provision best routes, ..

SDN vs legacy computer networks

- **Legacy computer networks**



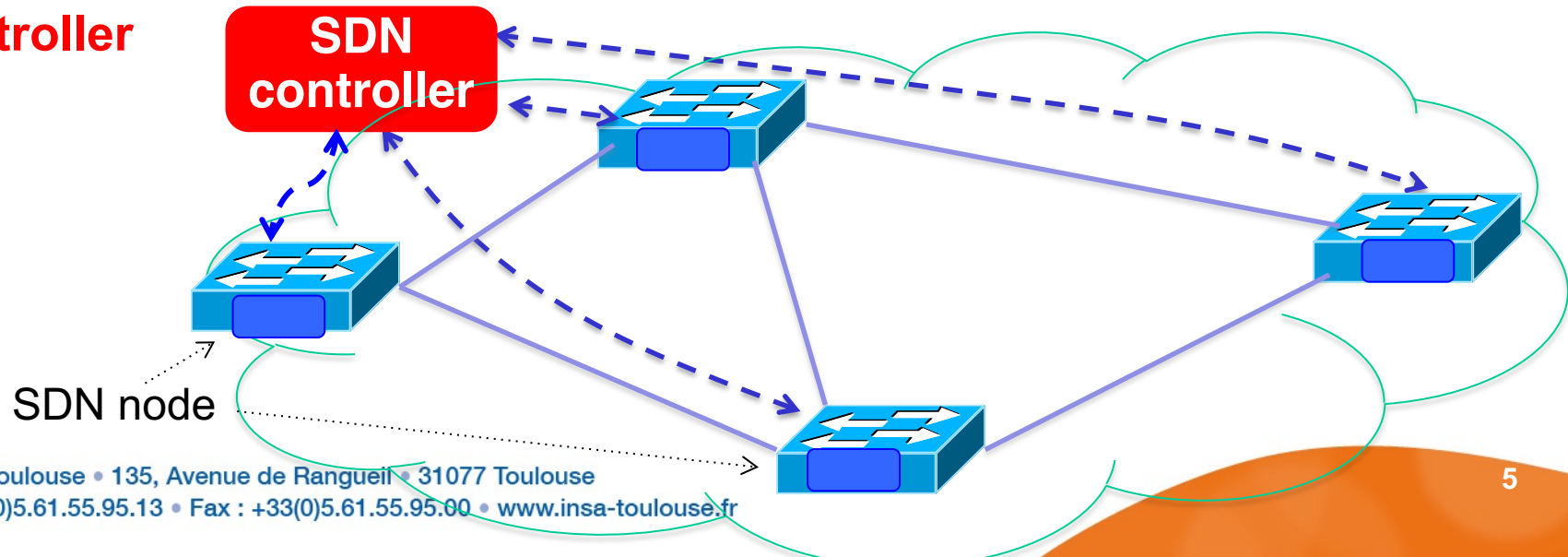
NMS : Network Management System

- **Management plane: network Functions related to network management : configuration, fault, performance& security mgt**

SDN vs legacy computer networks

- **What is SDN ?**
  - Many definitions, we adopt the one proposed by the ONF (Open Networking Foundation)
- **SDN design principles**
  1. Remove network control functions from switching devices =>
     - *Network Nodes =simple packet forwarders*
  2. Node programming & monitoring from one logical entity that runs on computer machines

=> **SDN controller**



SDN design principles

- **SDN design principles (continued)**

  2) network node programming from the SDN controller

  – *Following a flow based approach (as opposed to a destination based approach)*

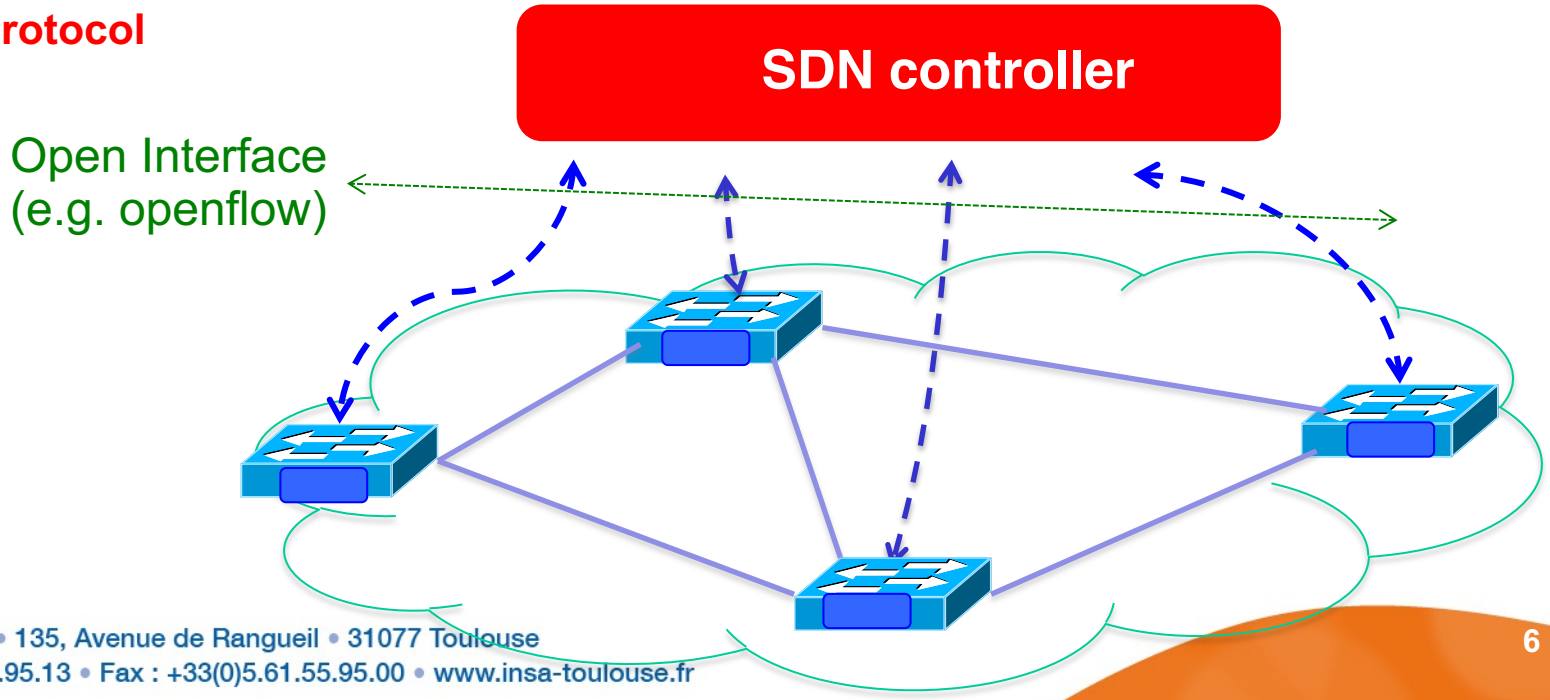
  ➢ High flexibility in the definition of a flow

  – ***On the fly***

  – *Via an **open interface***

  ➢ The de facto candidate :

  Openflow protocol

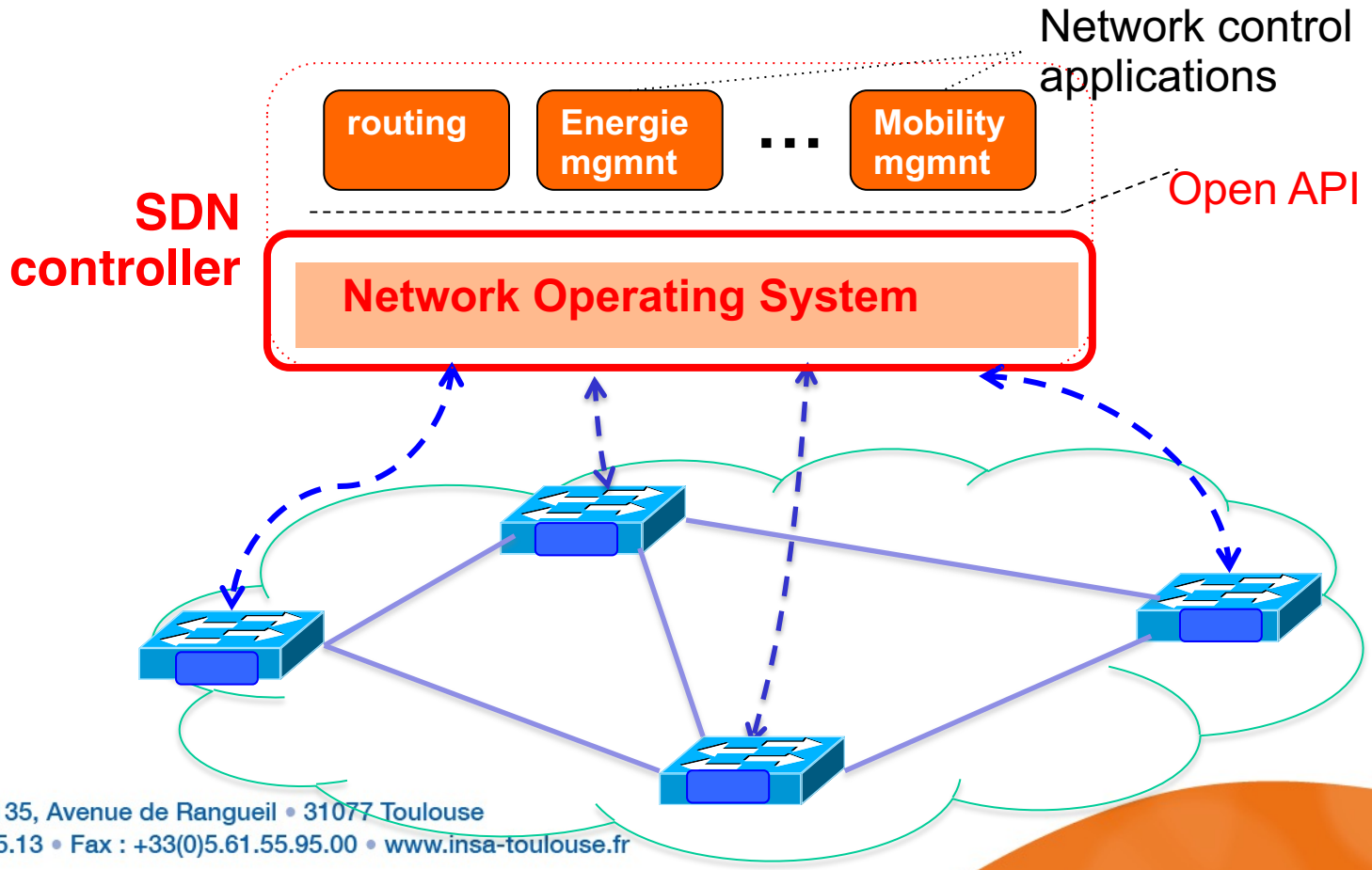


SDN design principles

3) network is programmable via **open API** exposed by the controller => **north-bound interface**

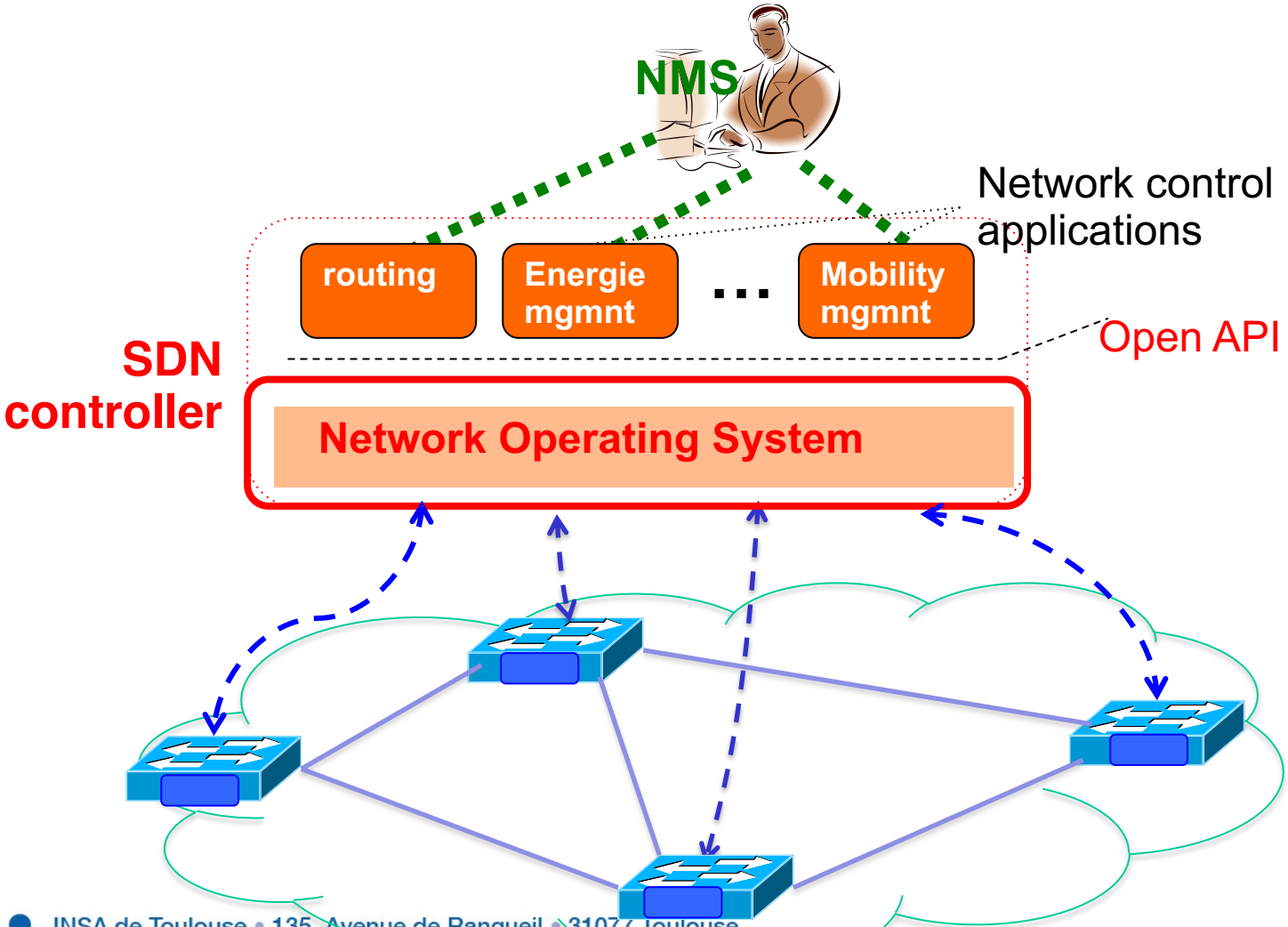
– *Controller implements a Network operating system that provides base abstractions, resources and services to ease network control*

➢ Topology discovery, etc.



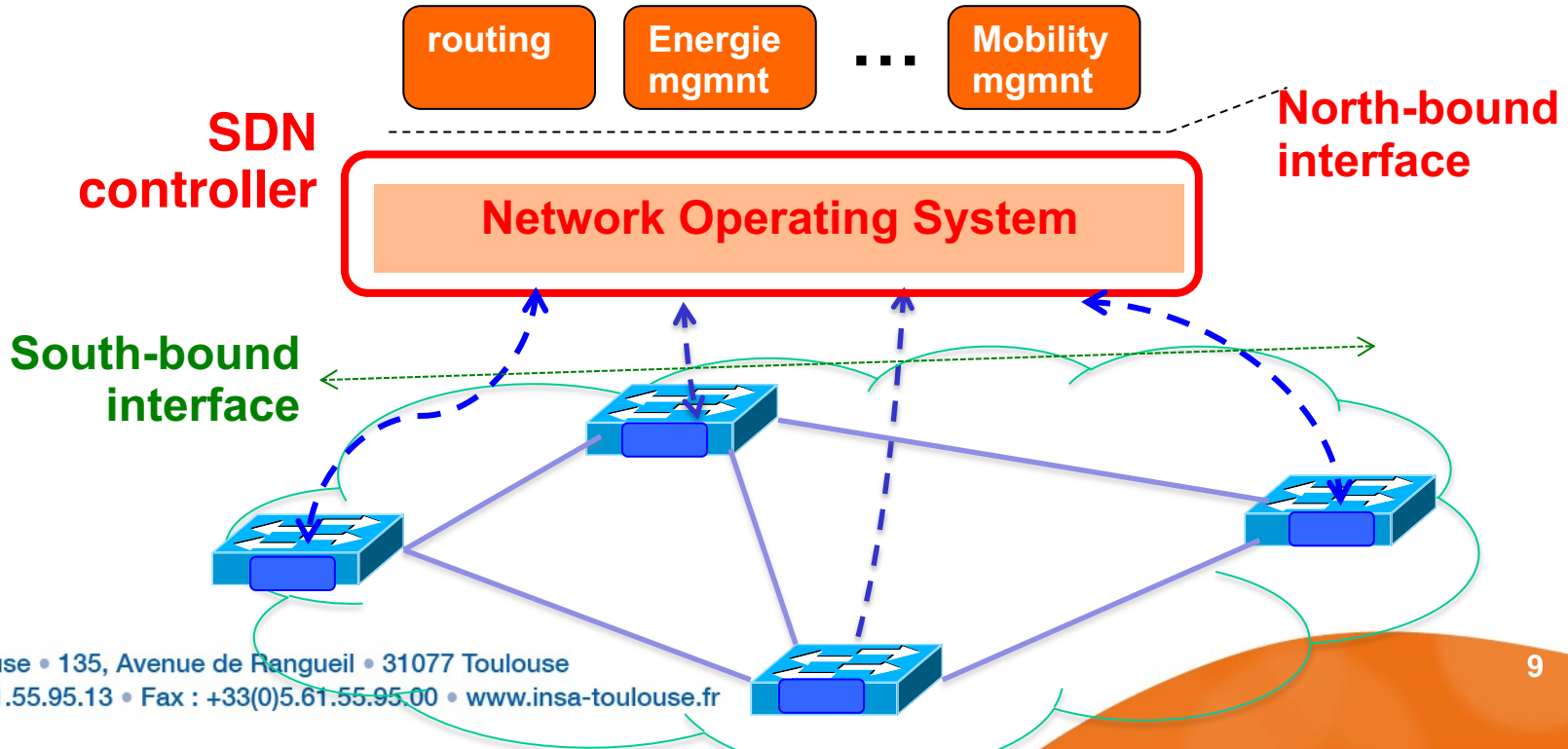
SDN design principles

- **What about network management functions ?**



SDN design principles

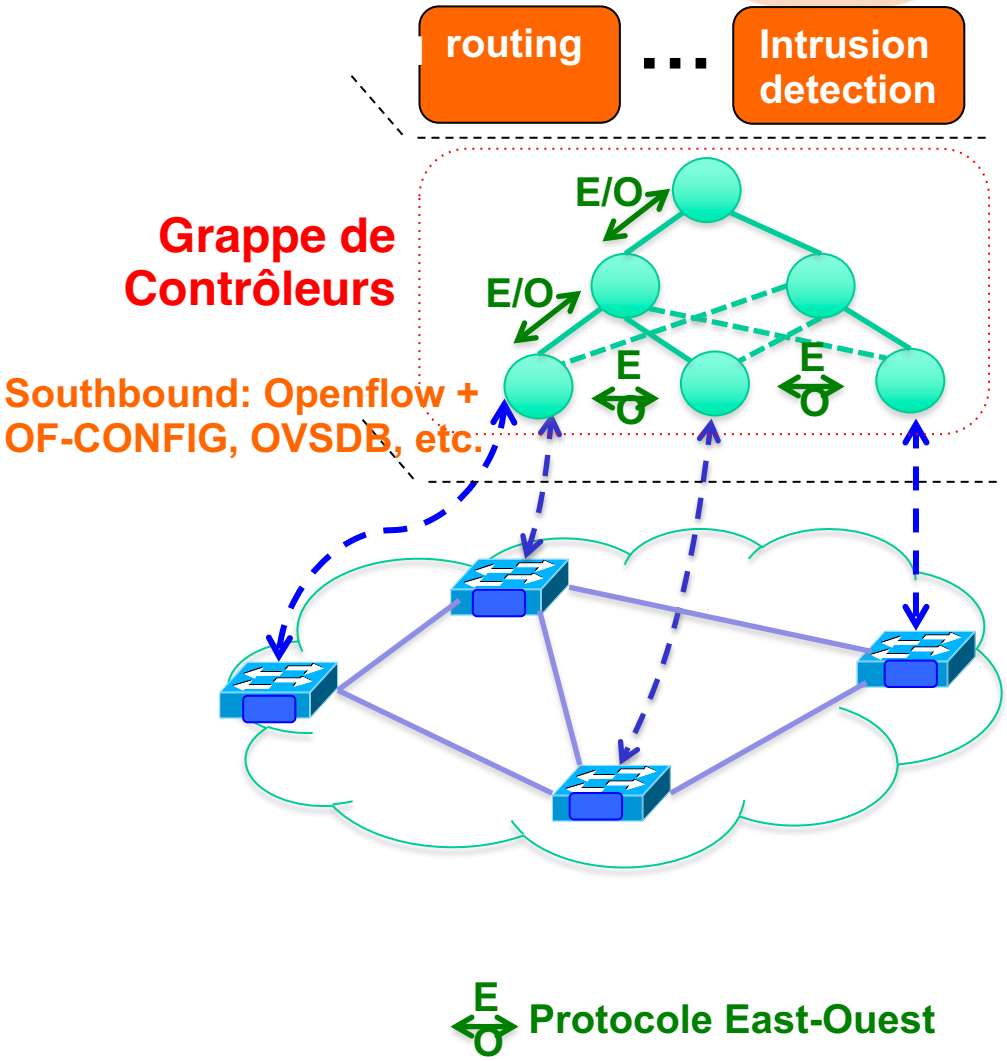
- **Many south-bound interfaces have been proposed:**
  - **Openflow,** POF, OnePK (Cisco), ForCES (Forwarding and Control Separation) de l'IETF, XMPP (Extensible Messaging and Presence Protocol) de Juniper, etc.
- **No agreed choice for the north-bound interface:**
  - Low-level and high level interfaces
    - *Some high level interfaces are even programming languages*



SDN ecosystem

- **Distribution of the control**
  - Scalability and availability
  - Hierarchical or fully distributed
  - East-Ouest protocol
- **Southbound interface:**

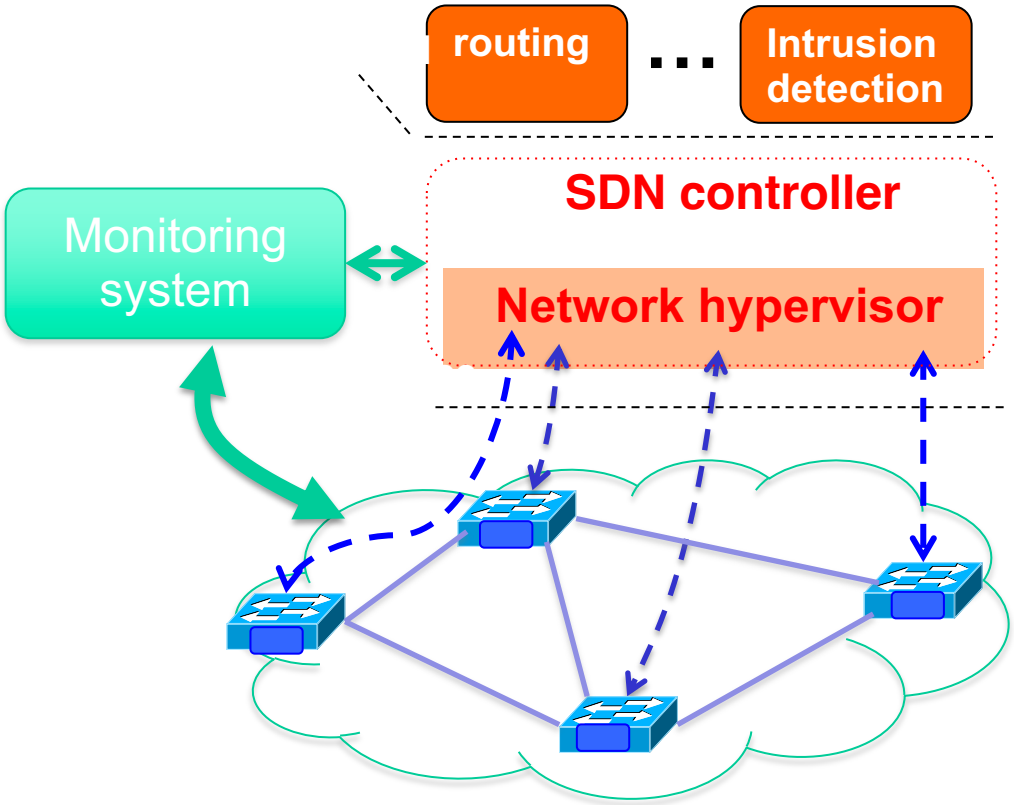
  lot of protocols +
  - Switch configuration protocols :
    - *NetConf, OF-Config, OVSDB, etc.*



SDN ecosystem

- **Monitoring system in addition to the monitoring capabilities of Openflow**
  - *To unload the SDN controller*

- **Network virtualisation**



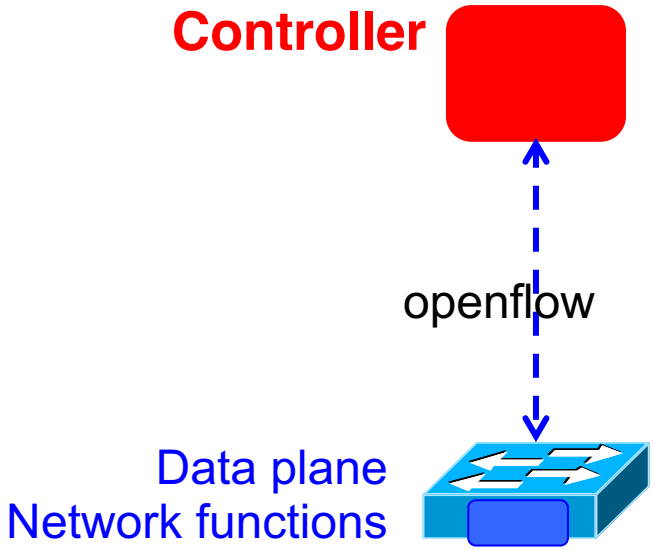
SDN benefits

- **Simplified network managment**
  - From the Network administrator point of view : master network control applications with little intervention on network devices
  - Orchestration and automatization and independence

  => Operational cost reduction
- **Fine-grained and dynamic control of network traffic**
- **Enables the emergence of novel network services**
- **Opens the network ecosystem to new actors**
  - With an expected reduction of investment costs

2. Brief introduction to Openflow

- **Openflow protocol used by the controller to program data plane network functions**



- **The protocol has and the capabilities of the controlled device are inter-dependent**

- **Design principles of Openflow :**
  - very simple data plane network functions
  - **Flexible** programming at the level of **flows** of packets

2. Brief introduction to Openflow

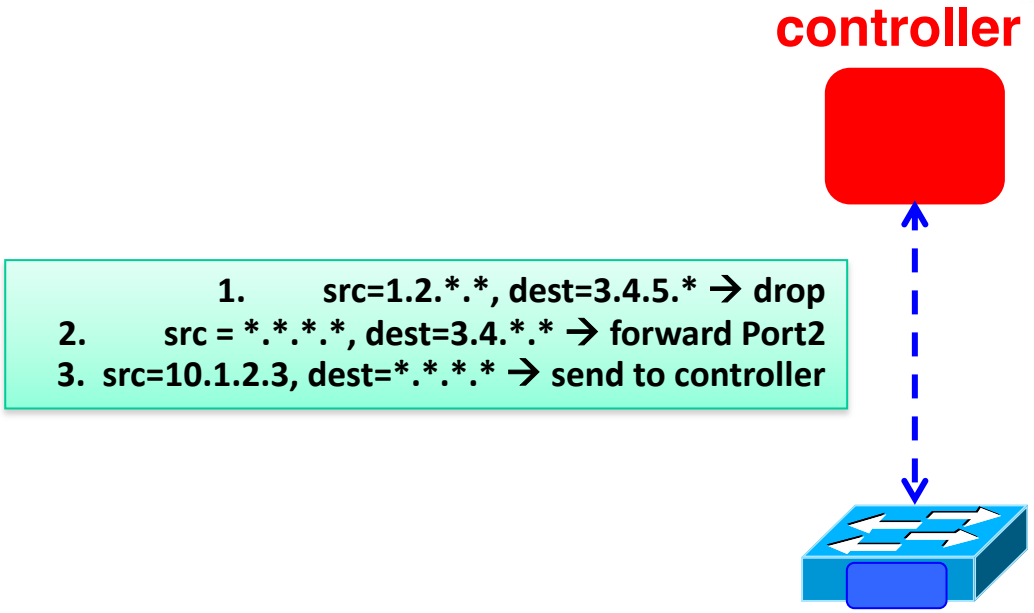
- **Roughly, on legacy devices, on a packet arrival**
  - router :
    - *match : the longuest IP network préfix*
    - *Action : forwards at the appropriate outgoing interface*
  - Layer 2 switch :
    - *match : MAC address*
    - *Action: forwards on one or all interfaces*
  - Firewall :
    - *match : IP address and/or UDP/TCP port numbers*
    - *Action : allows or blocks*
  - NAT :
    - *match : IP address or port number*
    - *Action : rewrites IP address or port number*
- **Whatever the network device : for each arriving packet, some bits of the header are matched to derive some predefined actions**

2. Brief introduction to Openflow

- **Openflow relies on this finding and proposes a programming scheme that inserts entries into the forwarding table that are related to a flow of packets**
- **An Openflow rule (entry of the forwarding table) composed of:**
  - **« matching rule » :** defines the flow of packets to which the rule applies by checking the values of some of its headers
    - *Example : VLAN-ID =25 & @IPsrce=192.168.1.* & @IPdest=192.168.*.1*
    - *Considers common protocol headers*
    - *High flexibility in the definition of the flow of packets*
  - **Actions** that apply to the flow of packets
    - *Switch to network interface x, remove, transmit to controller*
    - *rewrite the header (avec un masque), push/pop headers*
    - *Forward with a pre-defined rate*

2. Brief introduction to Openflow

- **Examples:**



- **These rules are installed in an Openflow node in a forwarding table called « flow table »**
  - An entry of a flow table

Openflow v1.0

- **Example of a flow table**

Port	Src MAC	Dst MAC	EtherType	VLAN ID	Priority	Src IP	Dst IP	IP Proto	IP ToS	Src L4 Port	Dst L4 Port	Action	Counter
*	OA:*	O1:*	*	*	*	*	*	*	*	*	*	P5	125
*	*	*	*	*	*	192.168.*.*	*	*	*	*	*	P3	5
*	*	*	*	*	*	*	192.168.1.*	*	*	*	6500	P1	15
*	*	*	*	*	*	*	*	0X1*	*	*	*	Controller	3

- **On a packet arrival : match its header to the flow table entry with the highest priority**
  - In case of match: update the counter and execute the actions
  - Ortherwise, remove the packet or send it to the controller

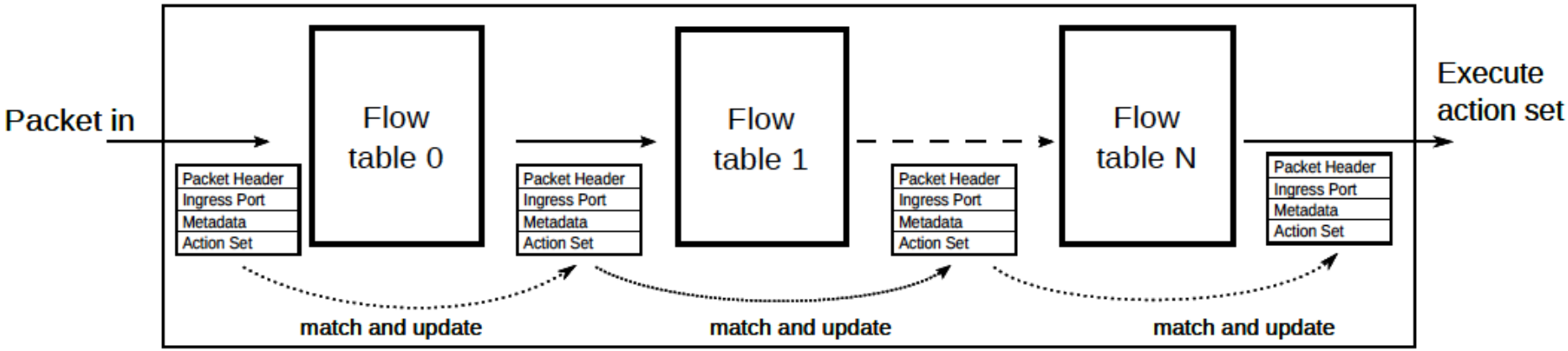
Openflow v1.0

- **Actions :**
  - Forwarding actions: Port, queue, All (interfaces), controller, Local, drop, etc.
  - Handle packets : add/remove VLAN-ID, change TOS, TTL, etc.

- **Counters :**
  - Per table : actives entries, matching count
  - Per flow : number/volume of matched packets, life-time
  - Per port : received/transmitted packets, error count
  - Per queue : number/volume of transmitted packets & overflow error count

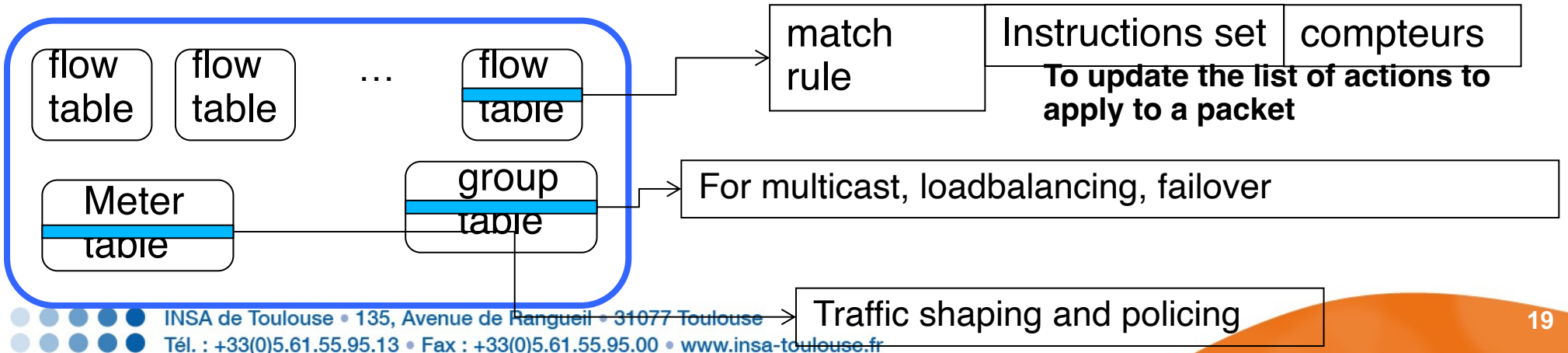
Openflow protocol versions

- **Main différences :**
  - Multiple flow tables in a switch (starting from version 1.1)



    - ***Introduction of the notion : Instruction and Actions-set + méta-data***
  - Openflow switch model

source : ONF



Instructions and actions lists

- **Instructions list :**

Instruction	Argument	Semantic
Apply-Actions	Action(s)	Applies actions immediately without adding them to the action set
Write-Actions	Action(s)	Merge the specified action(s) into the action set
Clear-Actions	-	Clear the action set
Write-Metadata	Metadata mask	Updates the metadata field
Goto-Table	Table ID	Perform matching on the next table

source : ONF

- **actions list (Openflow 1.4)**

actions	support	actions	support	actions	support
Output	O	Drop	O	**Set-Field**	o
Set-Queue	o	Group.	O	**Change-TTL (IP, MPLS, copy outwards/inwards)**	o
Push-Tag/Pop-Tag (VLAN, MPLS, PBB)			o		

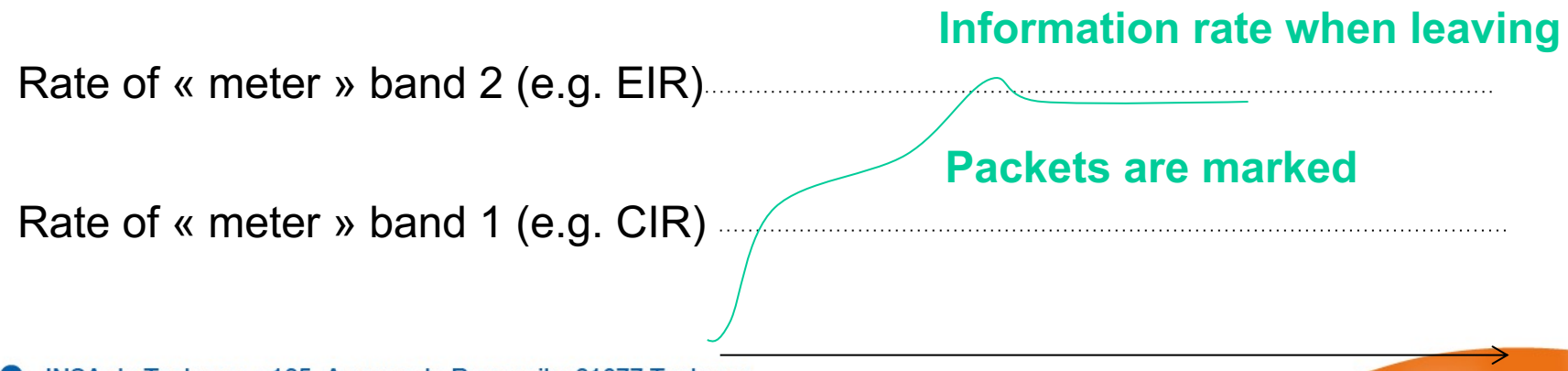
O: obligatoire, o: optionnel

Instructions and actions lists

- **Addition of a « group table » (from version 1.1)**
  - Each entry of the group table is a collection of lists of treatments on a flow of packets
    - *notion of « buckets » which is an ordered list of actions to apply to packets*
    - *A group table entry is typically composed of many buckets*
  - Each entry has a type (from the following list) : upon a packet match
    - *All : All the buckets are executed => Multicast/broadcast*
    - *Select : A single bucket is selected (according to its weight) and applied to the packet => load balancing*
    - *Indirect : the only predefined bucket is executed => can be pointed by many flow table entries*
    - *Fast failover : The first active bucket is executed => protection mechanism*

Liste des instructions et actions

- **Addition of a « meter table » (from version 1.3)**
  - Meter ?
    - *a switch component capable of measuring and enforcing the traffic of a flow of packets to a prespecified rate (a.k.a traffic regulator)*
    - *associated to flow table entries*
  - Composed of a set of « meter bands »,
    - *Each « meter band » has a rate parameter*
    - *Is active when the packet arrival rate exceeds its rate parameter to execute one of the following actions :*
      - ➢ Dropping the packets or remarking



Matchin rule& statistics

- **Matching rules apply to the following fields:**

	OF 1.0	OF 1.1	OF 1.2	OF 1.3 & OF 1.4
Ingress Port	X	X	X	X
Metadata		X	X	X
Ethernet: src, dst, type	X	X	X	X
IPv4: src, dst, proto, ToS	X	X	X	X
TCP/UDP: src port, dst port	X	X	X	X
MPLS: label, traffic class		X	X	X
OpenFlow Extensible Match (OXM)			X	X
IPv6: src, dst, flow label, ICMPv6			X	X
IPv6 Extension Headers				X

source : ONF

- **Statistics Related to:**

	OF 1.0	OF 1.1	OF 1.2	OF 1.3 & OF 1.4
Per table statistics	X	X	X	X
Per flow statistics	X	X	X	X
Per port statistics	X	X	X	X
Per queue statistics	X	X	X	X
Group statistics		X	X	X
Action bucket statistics		X	X	X
Per-flow meter				X
Per-flow meter band				X

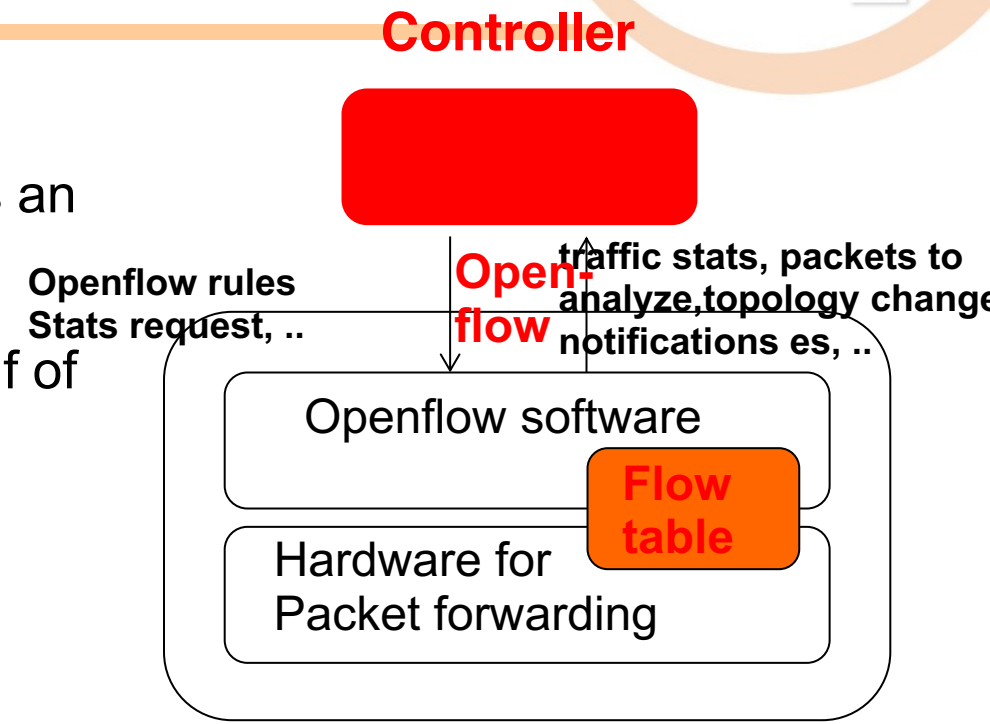
source : ONF

Openflow switches

- **Can be:**
  - Openflow-enabled (often)
    - *Legacy forwarding* +
    - *Openflow based forwarding*
  - Openflow-only
- **Hardware switches**
  - Different vendors: Legacy+newcomers
  - HP, brocade, Ciena, Cisoc, Juniper, NEC, IBM, Dell, Huawei, Netgear + Arista, **Pronto (Pica 8)**, Toraki, Quanta, Extreme summit, NetFPGA
- **Software switches**
  - **Open vswitch**, of13softswitch, Pantou, Indigo, LINC, XORPlus
  - Open vswitch : Implementation Open-source, potentiellement distribuée, d'un switch virtuel multi-protocoles et multi-couches

Openflow controllers

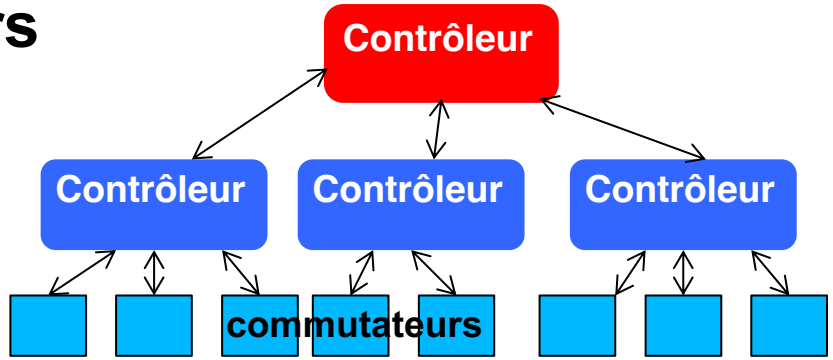
- **Openflow controller**
  - Implements a network OS that maintains an up to date view of the network
  - Programs the forwarding tables on behalf of Network applications

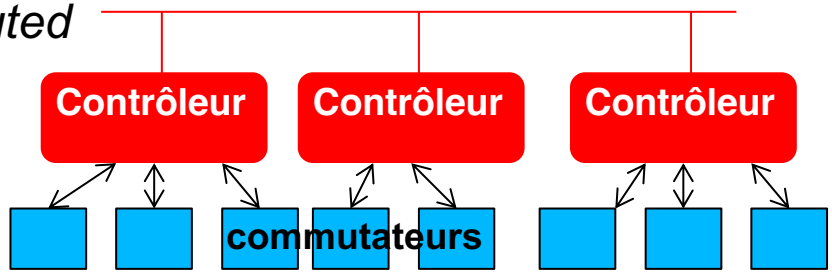


  - **North bound API** : usually REST API
  - **Network programming**
    - ***Reactive***
      - A first packet from a flow induces the insertion of a flow table rule => latency
      - Availability of the controller is crucial
    - ***Proactive***
      - Flow tables are populated prior to any transmission=>less flexibility but reduced latency
    - *Hybrid*

Openflow controllers

- **In production networks, the controller is a logical entity supported by many physical controllers**
  - Two main architectures
    - *Hierarchical*
    - *Distributed*





- **Many Openflow controllers**
  - commercial (supported by switch vendors) :
    - *Cisco ((Extensible Network Controller), HP (Virtual Application Networks), Nicira & Vmware (Network Virtualization Platform), NEC (Programmable flow controller), etc.*
  - Open-source

Openflow controllers

- **Many Openflow controllers (suite)**
  - Open-source

Examples	Programming language
Floodlight, Beacon, Maestro	Java
POX, RYU, pyretic	python
NOX, MUL, Trema	C/C++
Nodeflow	Javascript
Controllers used in production	**Programming language**
OpenDayLight	Java
ONOS	Java

Conclusion

- **Powerful concept with promising applications**
  - Enforcing complex & dynamic policies
  - Mobility management
  - Load balancing,
  - energy management,
  - Autonomic networking,
  - Network virtualization,
  - …
- **Adhesion to SDN principles but still some challenges**
  - Scalability
  - Distribution of the control
  - Security and availability